Real-life SOA Experiences and an Approach Towards Semantic SOA

Muhammad Ahtisham Aslam[1], Michael Herrmann[2], Sören Auer[1, 3] , Richard Golden[4]

[1] Betriebliche Informationsysteme, Universität Leipzig, Germany
{aslam,auer}@informatik.uni-leipzig.de
[2] DaimlerChrysler AG, Sindelfingen Germany
michael.hm.herrmann@daimlerchrysler.com
[3] Computer and Information Science Department, University of Pennsylvania, USA
auer@seas.upenn.edu
[4]Spirit Link GmbH, Germany
richard.golden@spiritlink.de

Abstract. Service Oriented Architecture promises the reuse of services. We recognized in an early stage of our SOA project, the gap between technical and semantical reuse of services. Thus, an agile Service Oriented Architecture (SOA) needs to be supported by ontologies. In this paper we introduce our real-life project and some already available results. We present a new 4-tier architecture to support Web services integration in semantic service oriented paradigm. The proposed 4-tier architecture is result of differentiation between architectural components (services) and those components that interact with services (orchestration). There is a further need for applying ontologies in order to describe services. Our research is based on a real-life project that measures the maturity of SOA and proves the need of semantics.

1 Introduction

In a Service Oriented Architecture (SOA) [1], the interaction between service providers and service consumers takes place in a loosely coupled way, where the service provider can also act as a service consumer. Web services (WS) [32] and its related standards like WSDL [9], SOAP [5], UDDI [7] and BPEL4WS [2] provide syntax based interaction and composition of WS in a loosely coupled way. Business logic can be modeled by using different process modeling techniques. For example, a value generating business system can be defined as a composition of value-generating activities in a Value Chain Diagram (VCD). Value chain activities are not isolated but affect each other. These activities are linked and data flow is defined between them to perform a specific operation with defined control and data flow. The Event Driven Process Chain (EPC) [31] is also a way out to compose methods together. EPCs are a method for representation of business process models. We can structure the control flow of a business process as a chain of events and functions by using EPC diagrams.

We are exploring SOA and WS based on real life requirements at Mercedes Car Group (MCG) in order to verify the vision of SOA. There is an overall strategy project coordinating every SOA activity at MCG. The technical reuse of services is solved by Web service standards. But, we recognized a gap of semantically reuse of services. Semantically reuse needs more than syntactical information – it needs a semantic description of services. Approaches like OWL-S [8], WSMO [11] and WSDL-S [12] are actually discussed in the community for this

purpose. OWL-S is suite of ontologies based on the W3C standard OWL [13] and consists of *Profile*, *Process Model* and *Grounding* ontologies. Some work is done in order to map BPEL to OWL-S semantic Web services [14, 33]. Several research papers have investigated different approaches for discovering and locating the semantically matching Web services [15, 16, 17].

This paper is organized as follows; the overall strategy project at MCG and some of its results are described in section 2. Semantic enhancements in the SOA are discussed in section 3. Section 4 describes the 4-tier architecture for Web services integration in semantic service oriented paradigm. Related work is discussed in section 5 and section 6 concludes and describes future directions for our work.

2 SOA activity at Mercedes Car Group (MCG)

The MCG Management decided to explore the vision of SOA and created a strategic project at MCG. We are going to name the work packages (WP), list the topics and describe some already available results of the project. We are also going to explore the need of semantics in order to reuse services. This overall strategy project is coordinating every SOA activity at the MCG. It coordinates but also supplements each single project (e.g. with a security concept in order to implement a common secure machine-to-machine communication). The project is cut into five working packages. The deliverable of WP1 is a well defined Reference Architecture; the deliverable of WP2 is a Development Landscape for projects; WP3 deliverables the hosting of SOA-projects; WP4 delivers the managing of Organization, Governance and Guidance; and WP5 delivers the modeling of business processes and adapting the Service-Oriented Modeling and Architecture (SOMA) [3]. A final result of the project is the positioning the strategic products in our enterprise. We have identified the following topics (random order) to verify the vision of SOA:

- Modeling processes in EPCs
- Tool based transformation from EPC to BPEL
- Deploying the process in the process engine
- Orchestrate services (exemplary)
- Communication of the services over a unified infrastructure
- Web services
- Implementing (at least) one registry
- Closer look at the top down approach
- Security
- Service Life Cycle Management
- Operational aspects like availability of each IT system and monitoring the infrastructure
- Explore the Enterprise Service Bus (ESB) [4]

We are going to describe requirements of the topic *Security* in this paragraph and already available results of the topics *Web services* and *Enterprise Service Bus* in the following paragraphs. There is a deeper look at security in sense of handling our four different parts of data classification specified in our policies: public, internal, confidential and secret. We expect experiences in supplier based security and Web services security in order to solve unlimited communication between different classified services. We focus on Host to Java communication and services that are hosted by a Host-System. Security aspects like authentication,

authorization, logging and classification of the data are well known challenges, which should be solved.

We had a closer look at Web service technology in past. Guidelines like using at least basic profile of WS-I standards, not publishing overloaded methods in Web services interfaces and creating WSDL first, then implement the service (contract first) have been established in our Enterprise. But, guidelines in such a dynamic technology have to be monitored constantly in order to adopt them.

We are dividing the topic ESB into concept of ESB and products of ESB. In small companies their concept may be solved by one product. In lager companies one product is not powerful enough to solve the requirement of integrating different platforms. Normally, larger companies have COBOL applications hosted on Systems like OS/390 or z/OS (legacy) and on the other hand J2EE application hosted in an Application Server. Thus, there is a need of a unified communication of those platforms, which can be solved by the concept of an ESB (fig. 1).

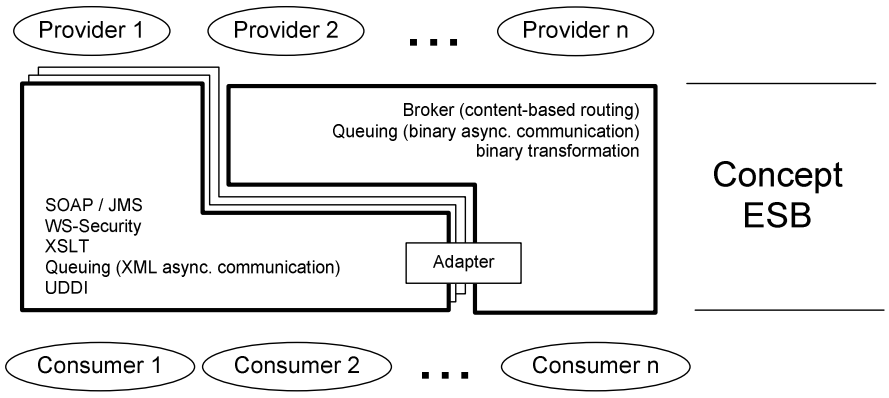

Fig. 1 enterprise-wide integration concept

The left side of figure 1 shows the traditional communication concepts to support the integration of legacy systems by binary messages. The right side of figure 1 shows the Web service based communication to integrate non-legacy systems like SAP, J2EE and .NET by XML messages. The two are connected (via adapter) to provide an enterprise-wide communication that combines features to support Web service technologies and the integration for messaging applications. The adapter integrates both massaging systems by mapping and transformation. The left side of fig. 1 is shadowed, because we expect more instances of supporting WS technology in future. More than one instance of the left side has to be integrated by an instance of the right side (broker).

In our ongoing strategy project at MCG we are going to verify the vision of SOA with available products of our suppliers. We recognized in an early stage of the project, the gap between technical (realized by WS standards) and semantical reuse of services. Semantically reuse needs more than technical information – it needs a semantic description of the services. The Chapters 3 and 4 outlined our ongoing lab based work in order to describe services semantically.

3 Semantics in Service Oriented Architecture

We are realizing a lab based applying of semantics into a business process, which is carefully selected from the project described in Chapter 2. We are going to outline the carefully selected process and the semantic approach in the following paragraphs. This EPC is the basis of our work in the lab: The "planning and execution" process is being executed each day in the morning in order to plan the assembly of the engines. The orders of assembling itself, assembling jobs and best batch sizes are going to be assigned. Therefore we need information about the availability of components, the need of engines and assembly restrictions like number of employees, capacity and availability of the equipment. The process flow is coordinating production, assembly, planer of assembly and scheduler/controller. The flow detects release order, determinates the need of engines, passes the plausibility check, calculates targets, passes restrictions check, creates production order, proves production order, and fixes the amount and the order of production. The following paragraphs describe the basis of our approach by adding semantics to EPC "planning and execution".

The goal is to model IOPE (inputs, outputs, preconditions, effects) [8] of each process step at the EPC "planning and execution", in the area of "power train", semantically. Additionally we are defining an automotive ontology based on the general concept of automobiles. We are adding specific constraints, instances of cars like an SLK (which is a Roadster) and the IT-systems needed to support the well selected process "planning and execution" described above. We are considering the maturity of OWL in regards to modeling the requirements of specific departments within MCG. We expect to gain significant experiences in modeling real-life requirements, in proving the maturity of OWL and in defining the restrictions of reusable enterprise ontologies. But, modeling IOPE's based on ontologies is not enough to realize a semantic SOA the lack of common understanding of services between the participants of a SOA and adding semantics in the SOA aims at providing shared meaning of business services within an organization and probably across the organizational boundary. Traditional SOA has three participants- *Service Provider*, *Service Requester* and *Service Registry* and semantic enhancements improve the role of the participants of SOA as:

- Semantic Web Service Provider
- Semantic Web Service Requester
- Semantic Web Service Registry

Semantic Web service provider can develop and advertise a Web service that provides its machine understandable meaning. Using a semantic Web services language can provide such machine understandable description of Web services. Three major candidates for semantic Web services (SWS) standards are OWL-S, WSDL-S and WSMO (fig. 2). The semantic Web service provider annotates services with domain ontologies to provide shared meaning of their Web service functionality by using any of these languages. Publishing SWS suppose that the service registry supports such SWS advertisements ([15]).

The requester of a WS is interested in finding a service that fulfills his functional requirements. A requester can find a service manually or can define a Web service request annotated with domain ontologies to provide request semantics (OWL-S Profile ontology). Such semantic requests can be used by computer agents to dynamically discover required services (e.g. [16] describes an approach to annotate and discover web services by matching semantics). The work discussed in [17] describes such an approach to automatically locate a Web service. Current UDDI structure supports only key word based searching of required

services. Such keyword based searching is inefficient and not precise because it finds those services also which are not performing the required functionality. Semantic enhancements in service registries demand more efficient mechanism to discover the required services on the basis of matching semantics. Locating the required services efficiently (semantically) is required by the semantic enhancements in the service registries. A good work has been done and is continuously improving the semantic base discovery of Web service by improving search algorithms [18] and enhancing the registry architecture. Chapter 4 describes a 4-tier architecture to address the integration issues raised with semantic enhancements in the SOA and Web services.

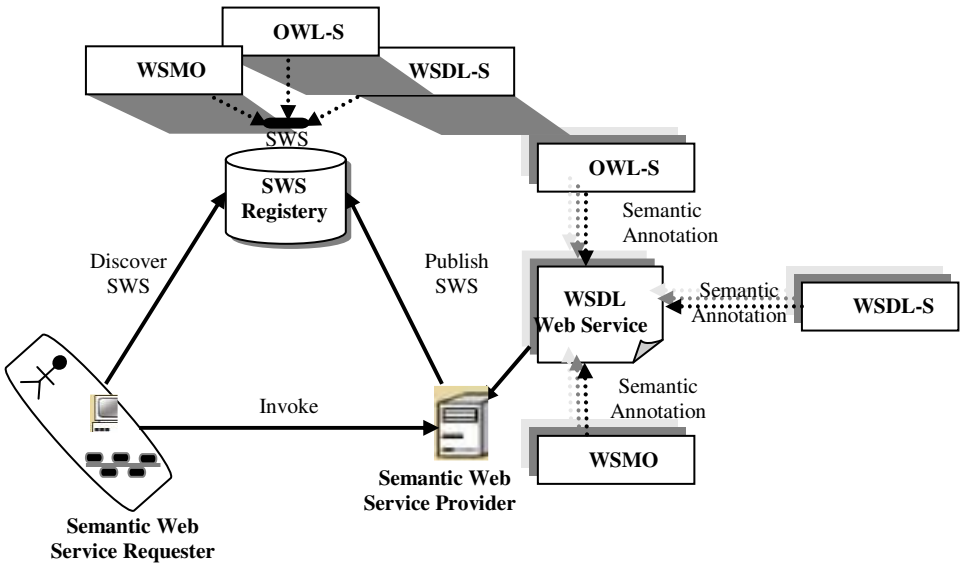

Fig. 2 Semantics based Service Oriented Architecture.

4 SOA and 4-Tier Integration Architecture

Chapter 4 is an approach in order to bridge the gap between real life SOA (chapter 2) and semantic Web services (chapter 3). SOA and semantics are considered from different communities. This section describes how 4-tier architecture helps to meet these integration issues raised with semantic enhancements in the SOA.

As the Web becomes more semantic and applications become more agile the need for an additional architectural layer becomes more prevalent. This new architectural layer choreographs the business rules and orchestrates the services by using ontologies. Figure 3 outlines how the choreography and orchestration layer (CO-Layer) and services layer in the "new" 4-tier architecture evolved from the business logic layer of the current 3-tier application integration architecture. This new architectural layer is not original. It is derived from the natural evolution of the business logic layer. The four layers in the proposed 4-tier integration architecture cooperate in order to provide the overall functionality. The invocation relationship

between the four layers is strict top-down invocation relationship. That is, components in upper layers invoke components in lower layers in order to accomplish their functionality and lower layers cannot invoke components in upper layers. This avoids circular invocation dependencies and ensures that the functionality separation is followed [19].

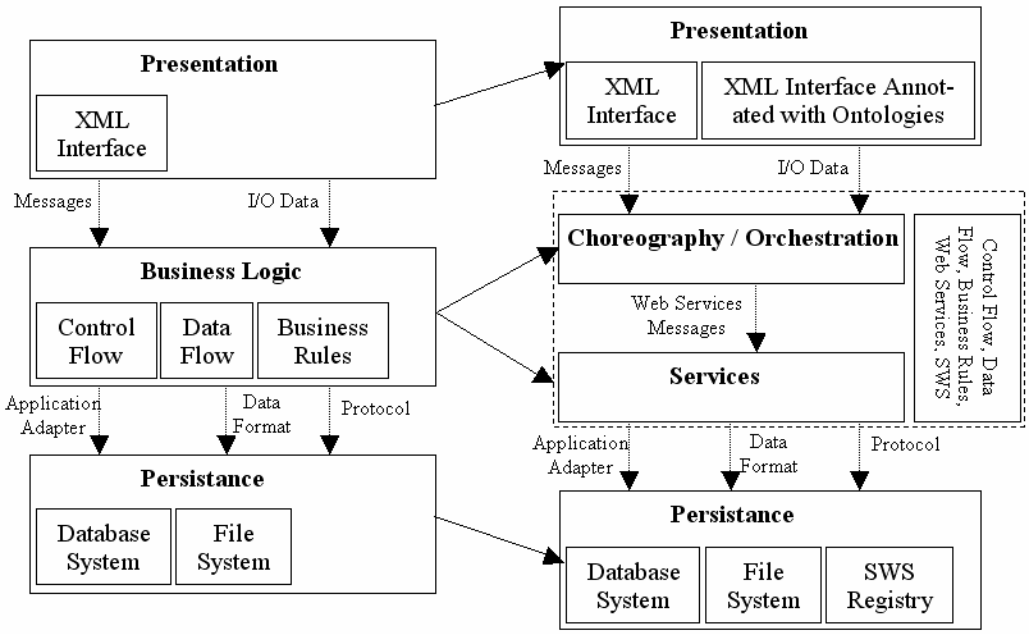

Fig. 3 4-tier semantic Web services integration architecture.

The 4-tier semantic Web services integration architecture consists of the following four tiers:

- Presentation Layer
- Choreography and Orchestration Layer (Business Logic Layer)
- Services Layer (Business Logic Layer)
- Persistence Layer

Presentation Layer provides interface to interact with integrated applications. Different interfaces can be provided to meet different integration requirements. For example XML provides a cross-platform standard for creating interface. It enables better reuse of user interface – presentation layer – in complex integration scenarios. Also, it can bypass some WSDL complexities when annotated with domain ontologies to provide data semantics. Different XML messages and input/output data creates the interface between presentation layer and business logic layer. Execution of business logic (process) is closely dependent on these messages and data bridging the co-ordination between presentation layer and business logic layer (CO-Layer and services layer). The resulted integrated application (service) can also present its interface (XML interface) for further co-operation with other services and applications. This interface can also be annotated with ontologies. Such a semantic interface helps in further dynamic and automated discovery, composition and invocation of these integrated services by semantic enabled systems. *Business Logic Layer* contains the components that implement the integration functionality. In traditional business application integration scenarios, business logic layer

define the control and data flow between the integrated applications and implement the business rules and business logic. Components in business logic layer interact with the preceding layer components by using some application adapter; data transport protocols and or formats (fig. 3). As long as business applications development trend has changed to business service development (extended with domain specific semantics) and the business logic layer comes in to focus, we begin to differentiate between architectural components that provide services and those components that either orchestrate services (service composition on the basis of matching semantics and aggregation) or choreograph them (business rules and workflow). This difference between components that realize specific use-cases (services) and components that organize those use-cases into dynamic business rules (choreography and orchestration) is emphasized by splitting the business logic layer in two. These two layers jointly play the same role as business logic layer (i.e. event management, process management, data management) and managing the control and data flow between services. When talking about integration architecture for SWS, we discuss the role of CO-Layer and services layer individually.

The *CO-Layer* choreographs and orchestrates the services in the service layer with business rules and semantics. This is the agile layer of the 4-tier software architecture model. This layer is required to be dynamic - to meet the changing requirements of the business enterprise. CO-Layer also needs to be adaptable as the enterprise grows through merger and acquisitions. Business logic and business rules can be implemented here by separating them from the underlying infrastructure of the system's operation. These rules can be implemented in some structure language. Even though CO-Layer and Services layer are emerged from business logic layer but components in these layers coordinate in such a way that these components are invoked precisely and in the right order by sending and receiving messages between CO-Layer and service layer components.

The *Services Layer* is not new. For many it remains equivalent to the business logic layer and contains a business rule service. The services layer is the realization of business processes in terms of discrete service definitions. This layer is inherently static as these services are tightly coupled to their implementations. However, when consistently defined in terms of IOPE with domain ontologies, services begin to reveal patterns of behaviour that can be modelled and orchestrated. As discussed above that the proposed integration architecture is a top down approach in which components in upper layer can invoke components in lower layer therefore, business logic layer can be interfaced with persistence layer by using some application adapters, protocols and data transport formats to exchange messages. Splitting the business logic layer in to CO layer and service layer results in an additional interface to query for semantic Web services and getting its response.

The reliability of the application integration architecture is intimately dependent on *persistence components (Persistence Layer)*. In the whole integration architecture database systems are used to store and to manage data. The data includes the messages, events, processes and configuration data. One possibility is to store all the information in some database but file systems can also be used to store data in files as part of persistence layer. The newly emerged layer "service layer" in the integration architecture added an additional component to persistence layer (i.e. SWS registry). The CO-Layer and services layer and persistence layer co-ordinate by sending query for a required semantic Web service (semantic Web service request) and getting its response (semantic Web service response).

The 4-tier semantic Web services integration architecture supports by integrating the semantically enriched Web services but it is not enough for dynamic, semi-automatic and automatic annotation, advertisement, discovery, selection, composition and execution of inter-organization business logic, making the Internet become a global common platform where

organizations and individuals communicate among each other to carry out various commercial activities and to provide value-added services [20].

5. Related Work

The work discussed in [20] addresses very well different areas about the semantic enhancements in Web services and dynamic discovery, invocation and composition issues. This work does not discuss about changes in application integration architecture with upcoming Web services technology. This issue is well addressed in our work. An implementation architecture for business information systems has been discussed in [22]. This work describes the n-layer information system architecture and role of each layer in an integration scenario. However, how these layers are interlinked with each other is not addressed in this paper. The n-layer architecture completely ignores WS and SWS technology. The success of the industrial and academic research related to the SOA and SWS depends on the success of ongoing efforts for the SWS development standards and how well its related issues have been addressed. Also, interoperability, collaboration and implementation of SWS related issues (as discussed above) and modeling and integration architecture is important in this domain.

So far there have been several efforts for the standardization of SWS technologies (e.g. WSDL-S, OWL-S, and WSMO). The WSDL-S proposal by the LSDIS Lab has been submitted to the W3C for standardization. The LSDIS Lab has also presented their work on annotating and publishing the WSDL services by using the WSDL-S [23, 24]. The LSDIS project team has also presented their work for dynamic discovery [25, 26] and composition [27] of these WSDL-S services. OWL-S is under discussion on the W3C platform and is part of the DAML program for the Semantic Web and the SWSs. The OWL-S 1.2 has been released at the time of writing this paper and consists of *Profile*, *Process Model* and *Grounding* ontologies. OWL-S supports compatibility between complex messages by using XSL Transformations [28]. The Web Services Modeling Ontology (WSMO) is another candidate to be declared as standard for the SWSs. The WSMO working group has also submitted their proposal to the W3C for standardization. The Web Services Modeling Language (WSML) is the language that formalizes the WSMO. The Web Services Execution Framework (WSMX) is the proposed architecture for the execution of the SWS. WSMO research community has also presented their work for semantic based discovery [29] and composition [30] of Web services defined with the WSMO.

6 Conclusion and Future Work

Rapidly growing technologies for loosely coupled and dynamic integration of business services demands for new architecture for integration scenarios. In this paper we have shown the topics of our real-life project and some already available results at MCG. We have also shown our plan to model services semantically based on a real-life business process outlined in chapter 3. We discussed how these processes could be annotated with business logic rules and constraints in some machine-readable workflow language. We discussed the annotation of these processes with domain ontologies to provide semantics of required services in defined workflow. We have finally presented a 4-tier integration architecture that addresses the integration issues raised with Web services and semantic Web services technologies in the SOA. We expect results of the

ongoing real-life project in October 2006 and results of our ongoing research activities in semantic SOA in 2007.

Acknowledgement: This work is partially supported by Higher Education Commission (HEC) of Pakistan (www.hec.gov.pk) under the scholarship scheme "Partial Support Scholarship for PhD Studies Abroad".

References

1. E. Ort: Service-Oriented Architecture and Web Services: Concepts, Technologies, and Tools. [online] Available http://java.sun.com/developer/technicalArticles/WebServices/soa2/.
2. Matjaz B. Juric, Benny Mathew and Poornachandra Sarang: Business Process Execution Language for Web Services. ISBN 1-904811-18-3, Packt Publishing Ltd, Birmingham, B27 6PA, UK
3. http://www-128.ibm.com/developerworks/webservices/ library/ws-soa-design1/
4. Keen, M. et al.: SOA with an Enterprise Service Bus in WebSphere Application Server V6 (IBM Reed Book SG, 2005, http://www.redbooks. ibm.com/redbooks/ pdfs/sg246494.pdf)
5. M. Gudgin, M. Hadley, N. Mendelsohn, J. Moreau and H. F. Nielsen: SOAP Version 1.2 Part 1: Messaging Framework. [online] Available http://www.w3.org/TR/soap12-part1/
6. K. Lawrence, C. Laler, A. Nadalin, R. Monzillo and P. Hallam-Baker: Web Services Security:OASIS Standard Specification, 1Feburary 2006. [online] Available http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf
7. K. Januszewski, E. Mooney, R. Harrah, S. Lee, J. Munter and C. V. Riegen: UDDI Version 3 Features List. [online] Available http://www.uddi.org/pubs/uddi_v3_features.htm
8. D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. Mcllraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan and K. Sycara: OWL-S Semantic Markup for Web Services. [online] Available http://www.daml.org/services/owl-s/1.2/overview/
9. R. Chinnici, J. Moreau, A. Ryman and S. Weerawarana: Web Services Description Language (WSDL) Version 2.0. [online] Available http://www.w3.org/TR/2006/CR-wsdl20-20060327/
10. T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic and S. Weerawarana: Business Process Execution Language for Web Services Version 1.1. [online] Available ftp://www6.software.ibm.com/software/ developer/library/ws-bpel.pdf
11. Web Services Modeling Ontology [online] Available http://www.wsmo.org/
12. R. Akkiraju, J. Farell, J.A. Miller, M. Nagarajan, A. Sheth and K. Verma : Web Service Semantics – WSDL-S [online] Available http://www.w3.org/2005/04/FSWS/Submissions/17/ WSDL-S.htm
13. D. L. McGuinness and F. V. Harmelen: OWL Web Ontology Language. [online] Available http://www.w3.org/TR/owl-features/
14. M. A. Aslam, S. Auer, J. Shen, M. Herrmann: Expressing Business Process Model as OWL-S Ontologies. In proceedings of the 2nd International Workshop on Grid and Peer-to-Peer based Workflows (GPWW 2006) in conjunction with the 4th International Conference on Business Process Management (BPM 2006), Vienna, Austria, LNCS 4103 , Sept. 4, 2006, pp.400-415.
15. M. Paolucci, T. Kawamura, T. R. Payne and K. P. Sycara: Importing the Semantic Web in UDDI. Proceedings of E-Services and the Semantic Web Workshop, 2002, pp. 225-236.
16. K. Sivashanmugam, K. Verma, A. Sheth, J. Miller: Adding Semantics to Web Services Standards. Proceedings of the 1st International Conference on Web Services (ISWC'03), Las Vegas, Nevada, June 2003, pp. 395-401.
17. U. Keller, R. Lara, H. Lausen, A. Polleres and D. Fensel: Automatic Location of Services. Proceedings of Second European Semantic Web Conference (ESWC 2005), Heraklion, Crete, Greece, May/June 2005, LNCS 3532, pp. 1-16.

18. N. Srinivasan, M. Paolucci, K. Sycara: An Efficient Algorithm for OWL-S Based Semantic Search in UDDI. Proceedings of First International Workshop on Semantic Web Services and Web Process Composition, SWSWPC 2004, San Diego, CA; USA; July 2004, LNCS 3387, pp. 96-110.
19. Christoph Bussler: B2B Integration Concepts and Architecture. ISBN 3-540-43487-9, Springer-Verlang Berlin Heidelberg New York.
20. J. Cardoso, A. Sheth: Introduction to Semantic Web Services and Web Process Composition. In proceedings of the First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004), San Diego, CA, USA, July 2004, LNCS 3387, pp. 1-13, 2005.
21. V. Tosic, B. Pagurek, K. Patel, B. Esfandiari and W. Ma: Management Applications of the Web Service Offerings Language (WSOL). Proceedings of 15th International Advanced Information Systems Engineering Conference, CAiSE 2003 Klagenfurt/Velden, Austria, June 2003, LNCS 2681, pp. 468-484.
22. H. P. Steiert: Towards a Component-based n-Tier C/S-Architecture. Proceedings of the Third International Workshop on Software Architecture (ISAW), Orlando, Florida, United States, ISBN 58113-081-3, pp. 137-140.
23. R. Akkiraju, J. Farell, J.A. Miller, M. Nagarajan, A. Sheth and K. Verma : Web Service Semantics – WSDL-S [online] Available http://www.w3.org/2005/04/FSWS/Submissions/17/WSDL-S.htm
24. A. Patil, S. Oundhakar, A. Sheth, K. Verma: METEOR-S Web service Annotation Framework. Proceeding of the 13th International World Wide Web Conference, July 2004, New York, NY, USA, July 2004, ISBN: 1-58113-844-X, pp. 553-562.
25. P. Rajasekaran, J. Miller, K. Verma and A. Sheth: Enhancing Web Services Description and Discovery to Facilitate Composition. Proceedings of First International Workshop on Semantic Web Services and Web Process Composition, SWSWPC 2004, San Diego, CA; USA; July 2004, LNCS 3387, pp. 55-68.
26. K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar and J. Miller: METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services. Journal of Information Technology and Management 2004.
27. R. Aggarwal, K. Verma, J. Miller, W. Milnor: Dynamic Web Service Composition in METEOR-S. Technical report, LSDIS Lab, Computer Science Department, UGA, 2004.
28. http://www.w3.org/TR/xslt
29. M. Stollberg; U. Keller;D. Fensel: Partner and Service Discovery for Collaboration Establishment with Semantic Web Services. In Proceedings of the Third International Conference on Web Services, Orlando, Florida, July 2005, ISBN 0-7695-2409-5, pp. 473-480.
30. D. Sell, F. Hakimpour, J. Domingue, E. Motta and R. C. S. Pacheco: Interactive Composition of WSMO-based Semantic Web Services in IRS-III. In proceedings of the First AKT Workshop on Semantic Web Services (AKT-SWS04) KMi, The Open University, Milton Keynes, UK, December 8, 2004.
31. Scheer, A.-W.; Thomas, O.: Geschäftsprozessmodellierung mit der ereignisgesteuerten Prozesskette, in: Das Wirtschaftsstudium 34 (2005), Nr. 8-9, S. 1069-1078
32. W3C Web Services Activity; http://www.w3.org/2002/ws/
33. M. A. Aslam, S. Auer, J. Shen: From BPEL4WS Process Model to Full OWL-S Ontology. In proceedings of Posters and Demos 3rd European Semantic Web Conference (ESWC 2006), Budva, Montenegro, June 11-14, 2006, pp. 61-62.